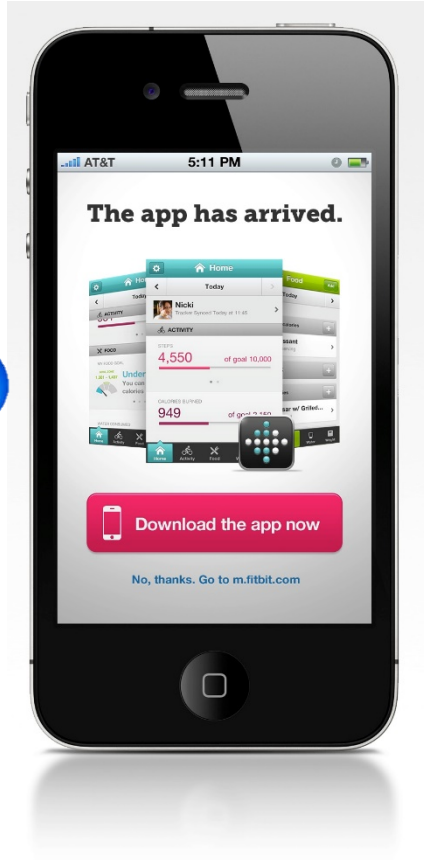
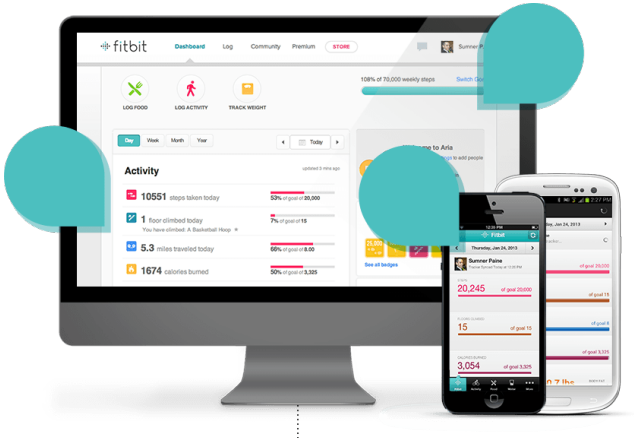
The fabryq IoT prototyping platform

Will McGrath
PhD Student

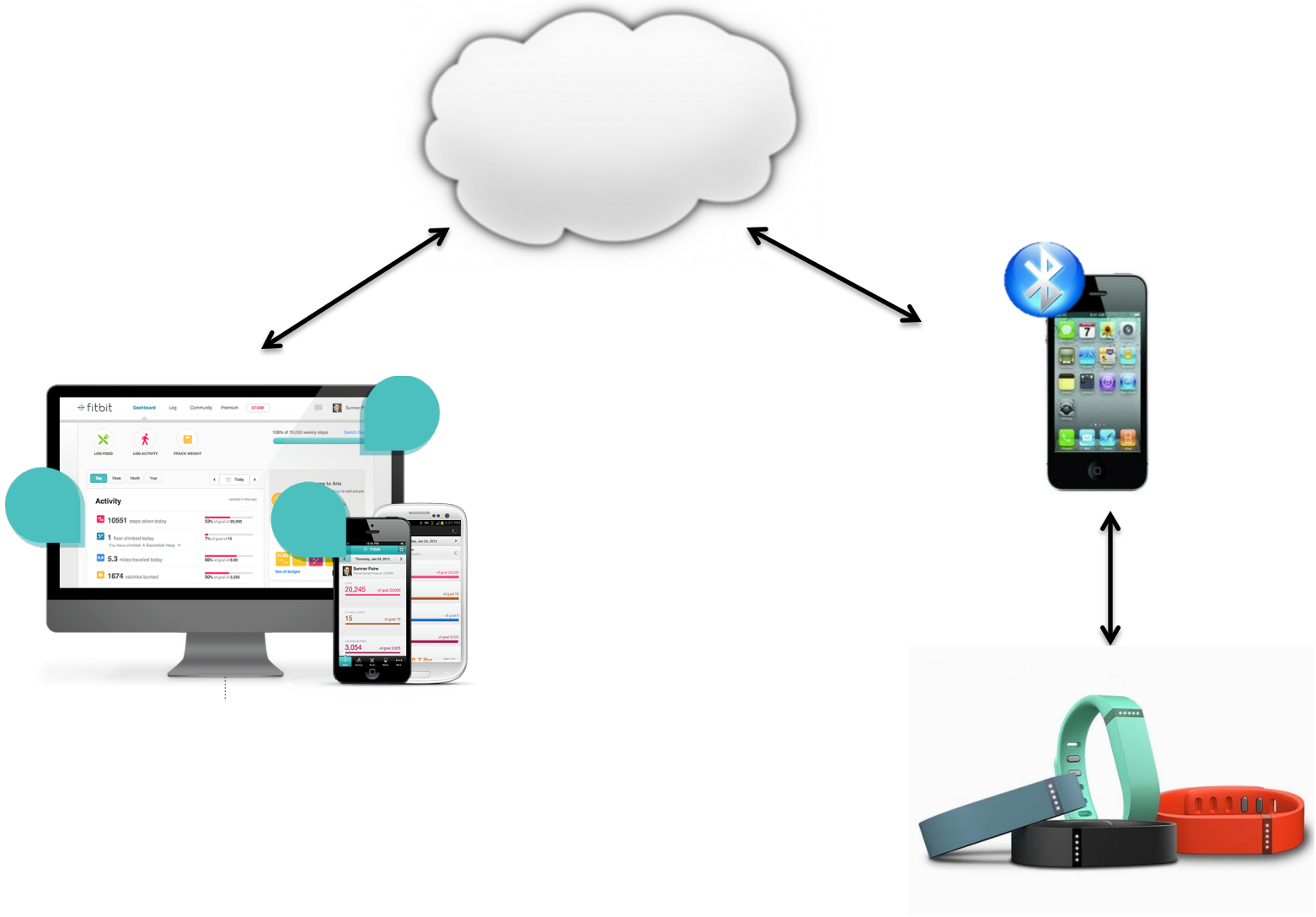
Advised by Björn Hartmann

Motivation


AT&T
5:11 PM
The app has arrived.
Download the app now
No, thanks. Go to m.fitbit.com


fitbit
Activity

Fitness Tracker



Embedded Gateway Cloud

- Embedded
  - Low Power device with sensors
  - Short range wireless
- Gateway
  - Bridge between short and long range
- Cloud
  - Access over Internet
  - Stores, aggregates, and provides data

Challenges

- Writing code in several languages (Server, Mobile, Web, etc.)
- Dealing with networking asynchronicity
- Hard to quickly prototype or iterate ideas

fabryq

- Platform to allow rapid prototyping of applications for new low-power connected devices
- Allows JavaScript web apps to communicate with smart devices via Bluetooth Low Energy (BLE)

Bluetooth Low Energy

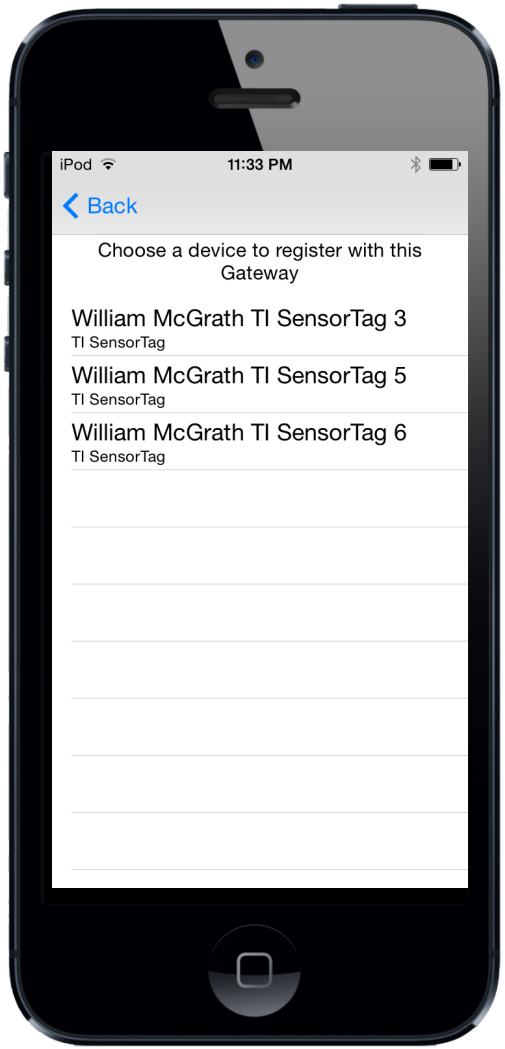
- Low power short-range protocol
- Widespread support
- Growing number of devices

BLE Basics

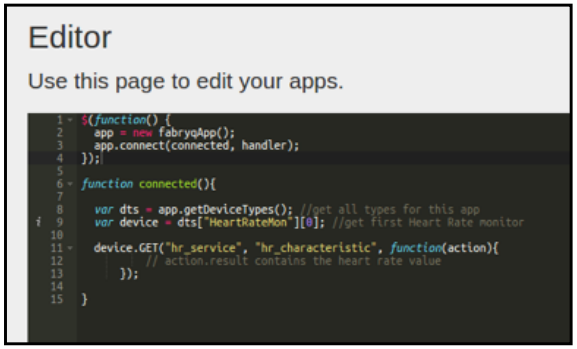
- Variables arranged in a table (GATT)
- Organized by services and characteristics
- Accessed using commands
  - SET (Write)
  - GET (Read)
  - NOTIFY (Read on change)
- “Star” network topology

App Development

- BLE devices and apps must be registered with fabryq before use
- Applications specify necessary devices

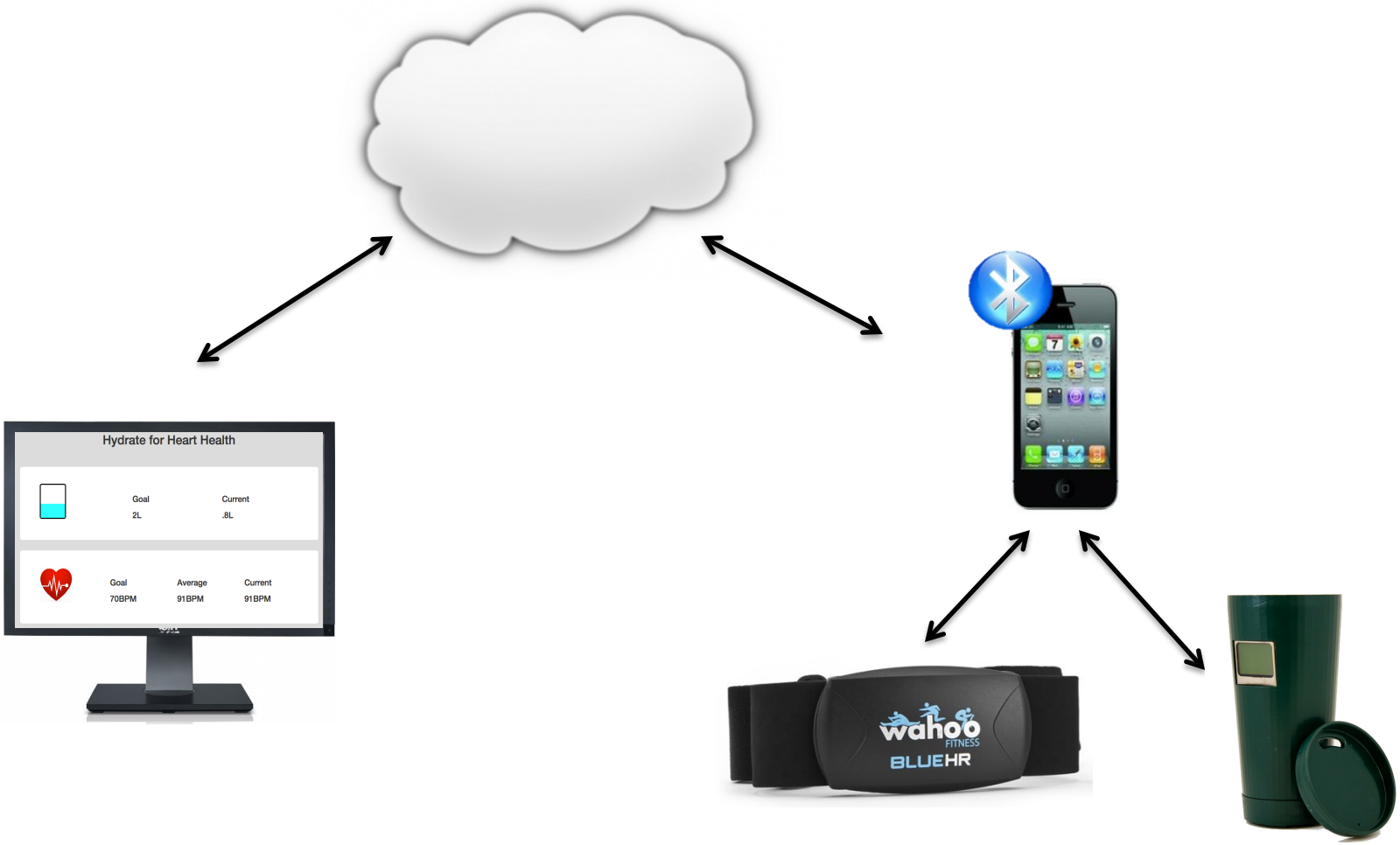


App Development

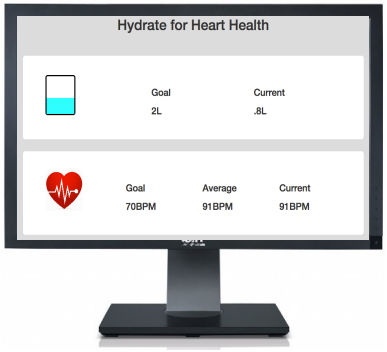


A **Register devices** by demonstration → **B** **Define application** What kind of devices? → **C** **Write Fabryq.js Code** → **D** **Run Application** in web browser or on phone

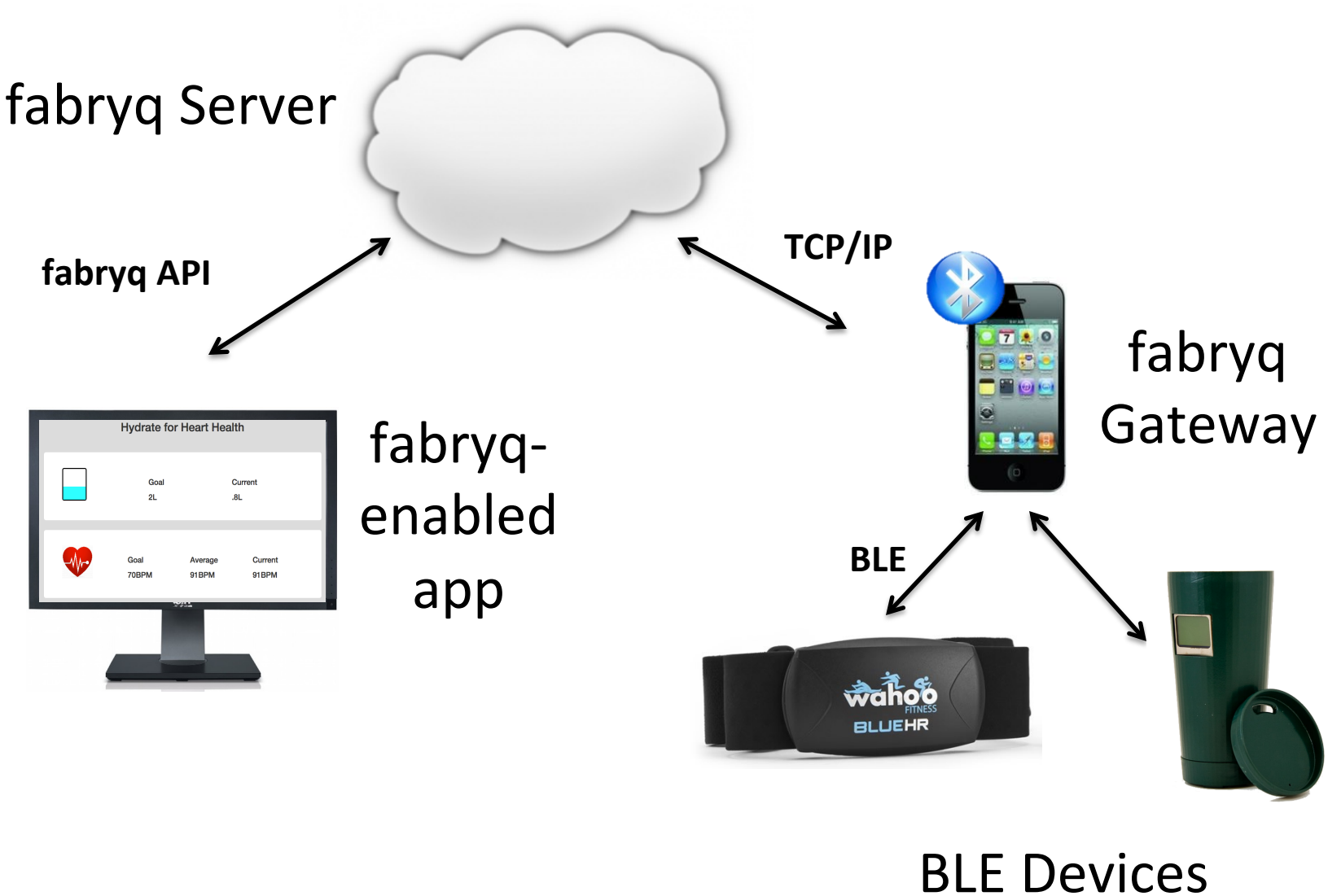
Example Application



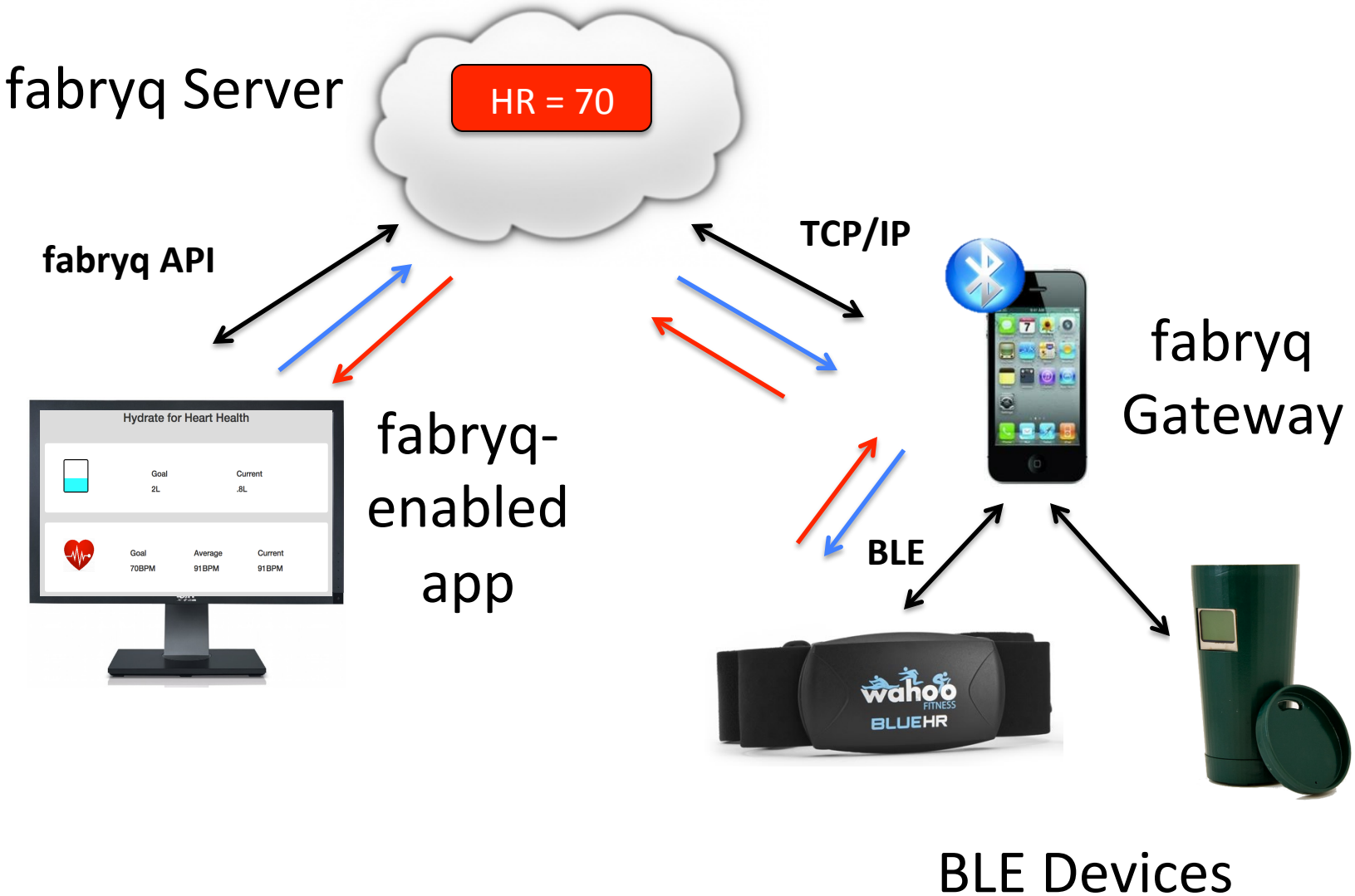
Example Application



Overview

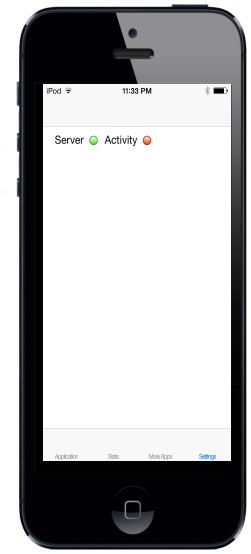

fabryq Server
fabryq API
TCP/IP
Hydrate for Heart Health
Goal
2L
Current
.8L
Goal
70BPM
Average
91BPM
Current
91BPM
fabryq-enabled app
fabryq Gateway
BLE
wahoo
FITNESS
BLUEHR
BLE Devices

Fabryq - Commands



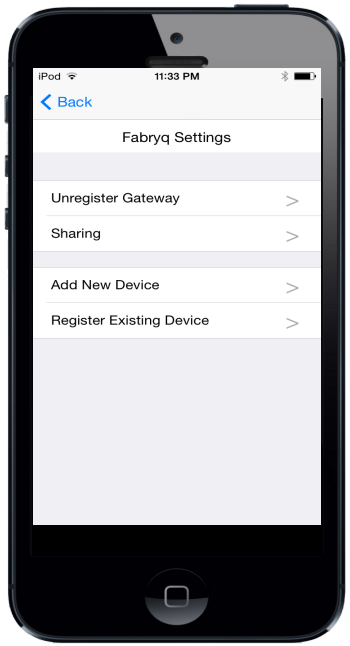
Gateway

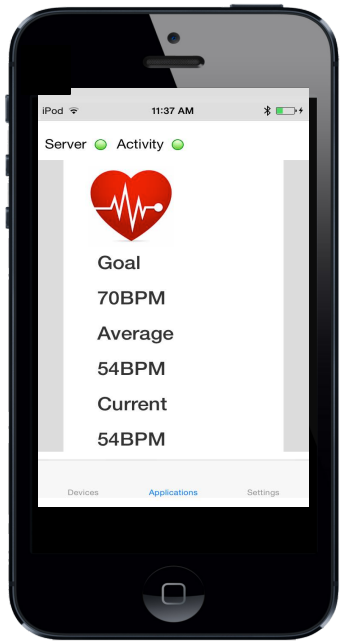
- Hardware that enables a device to act as a bridge between BLE and the Internet
- Fabryq currently supports iOS and OSX gateways



fabryq Gateway Features

- Allows users to manage associated devices
- Display custom user interfaces





Gateway Challenges

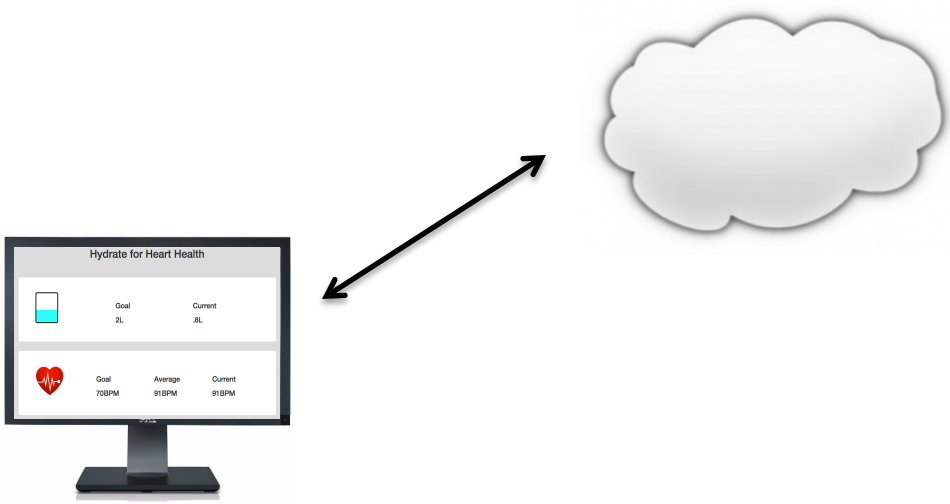
- Backgrounding
- Device identification/management
- Power usage tradeoff

Server

- Stores, aggregates, and provides data
- Supports long-running apps (Agent)
- Manages
  - Commands
  - Users
  - Devices
- Scaling can be challenging

fabryq API Goals

- Make getting data as simple as possible
- Allow for flexibility in program structure
- Permit long running applications



fabryq API Implementation

- Protocol proxy (GET, SET, NOTIFY)
- Receiving events
  - Command completion
  - System / error conditions
- Spawning agents

Sporadic vs Continuous

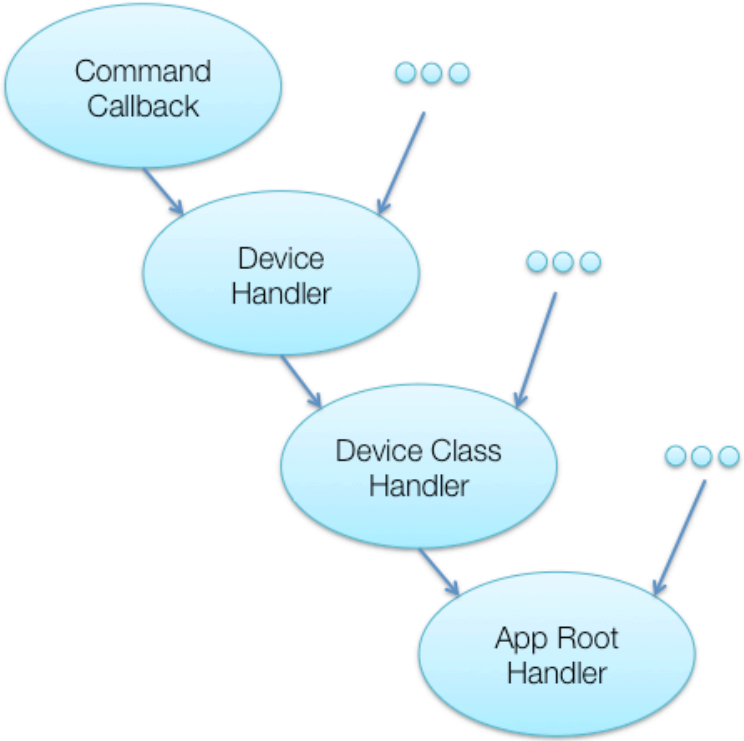
- Two types of supported applications
  - Sporadic – Infrequent usage and communication with device (i.e. light control)
  - Continuous monitoring – Application logic is constantly running waiting to act on events (i.e. security system)

Feature Breakdown

- Sporadic
  - UI on gateway or Web
- Continuous
  - Client UI launches/controlls agent application running constantly in cloud
  - Agent can push notifications to user

Event Handling

- Tree-shaped event and error handling structure
- Supports both callback-based and Swing-like handling



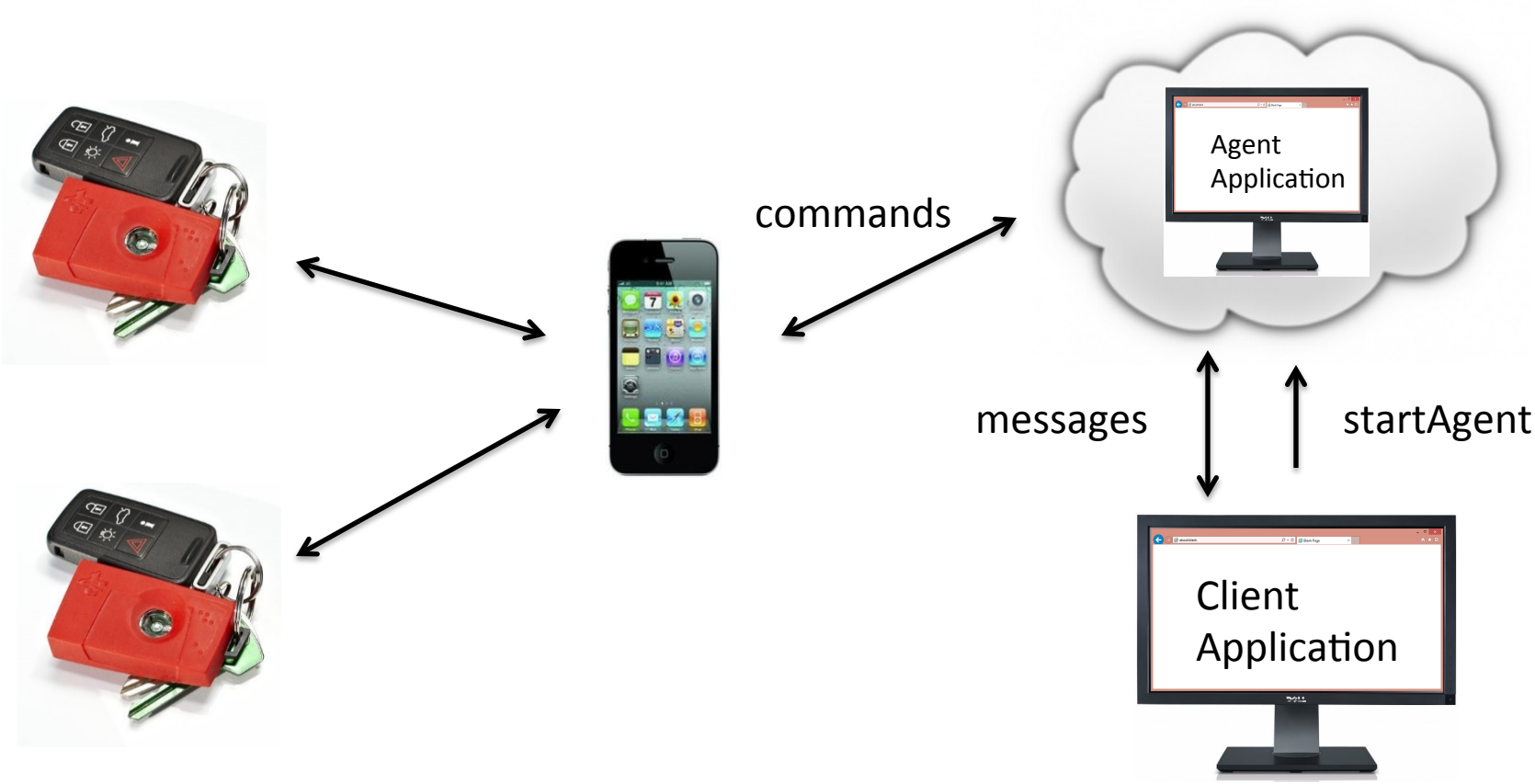
ID aliasing

- Hex parsing for known devices
- Extensible json format

```
"suuid":"FFE0",
"characteristics":[
        {"name":"button_value",
        "cuuid":"FFE1",
        "format":[
                {"name":"button",
                "start_byte":0,
                "end_byte":0,
                "order":"LSB",
                "converter":pass_through}]}
        ],
"characteristics_hash":{}
    },
```

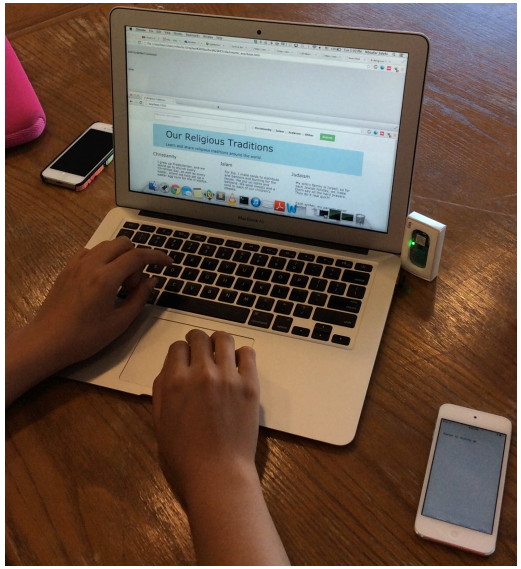
Agents

- Agent support for long-running applications

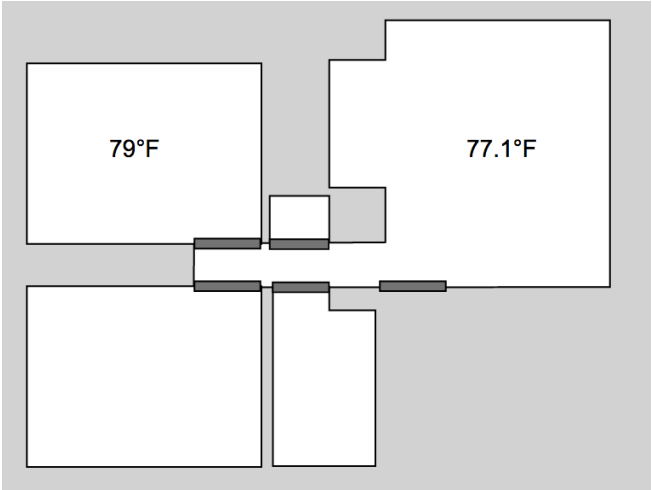


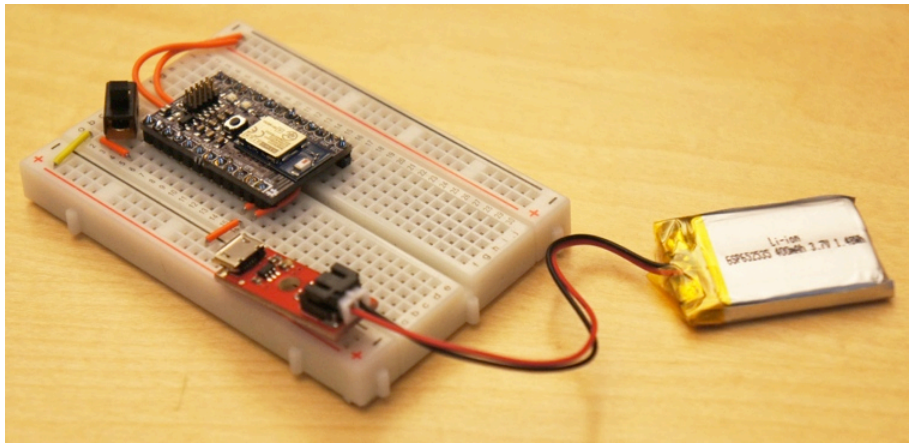
Evaluations

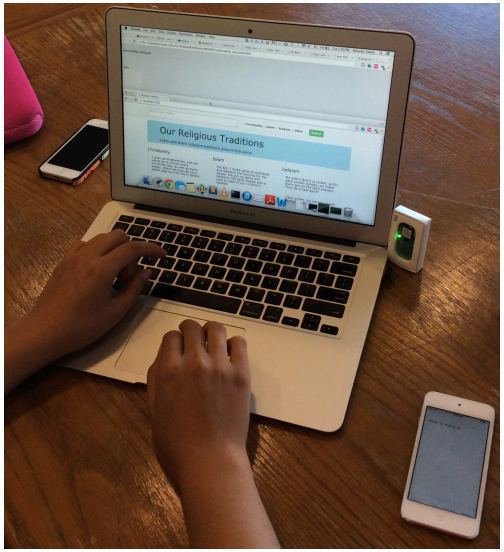
- ufabryq hackathon
- Lightning hack sessions
- Class prototype



Example Apps

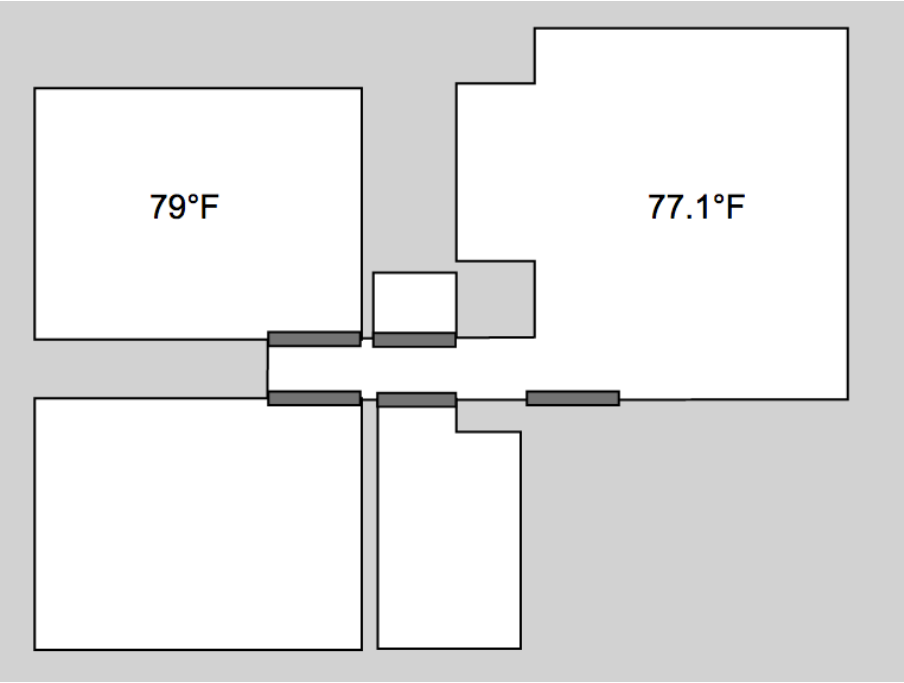






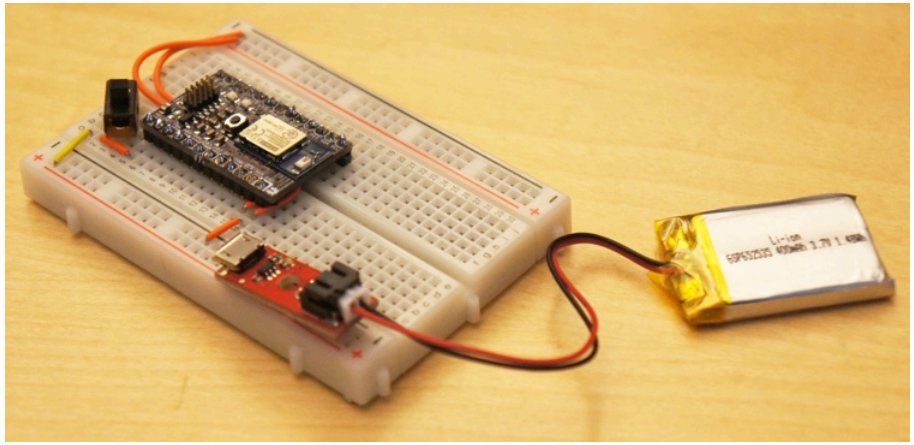
Home Monitor

- Easy to stream data (~1s latency)
- Creating the rest of app is time-consuming



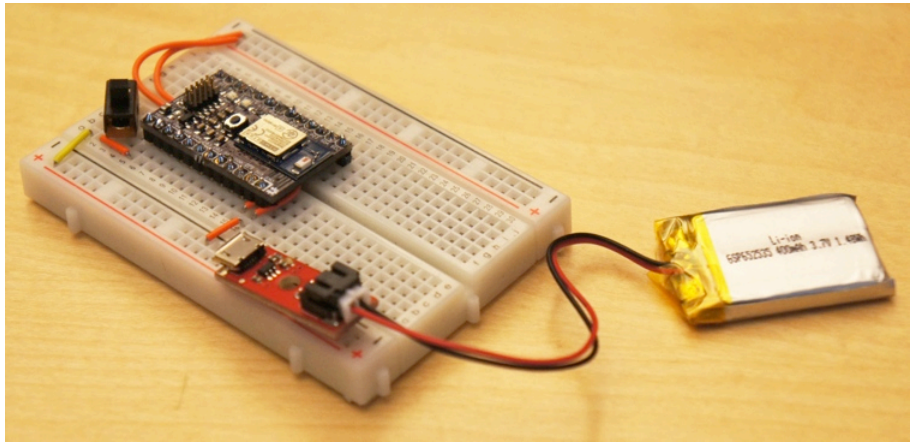
µfabryq

- All JavaScript hardware prototyping
- Based on BLE113
- Arduino-like Javascript commmands



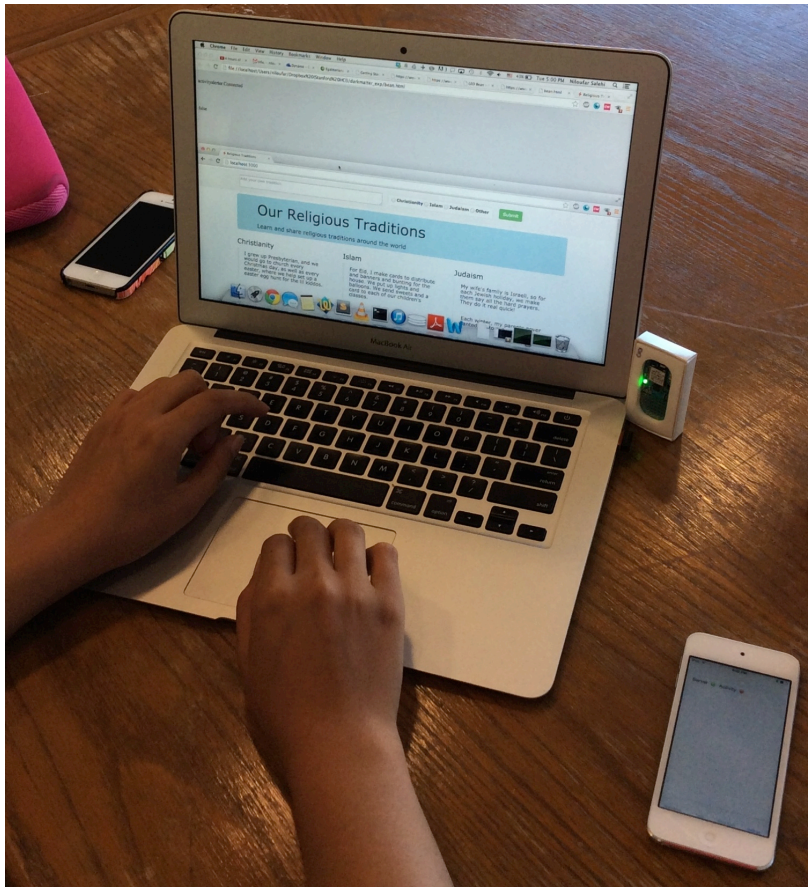
µfabryq

- Access to
  - GPIO
  - ADC
  - PWM
  - Interrupts
- Limited by latency and hardware



Page Served Indicator

- Easy to get running
- Easy to integrate with existing services (JavaScript or REST)

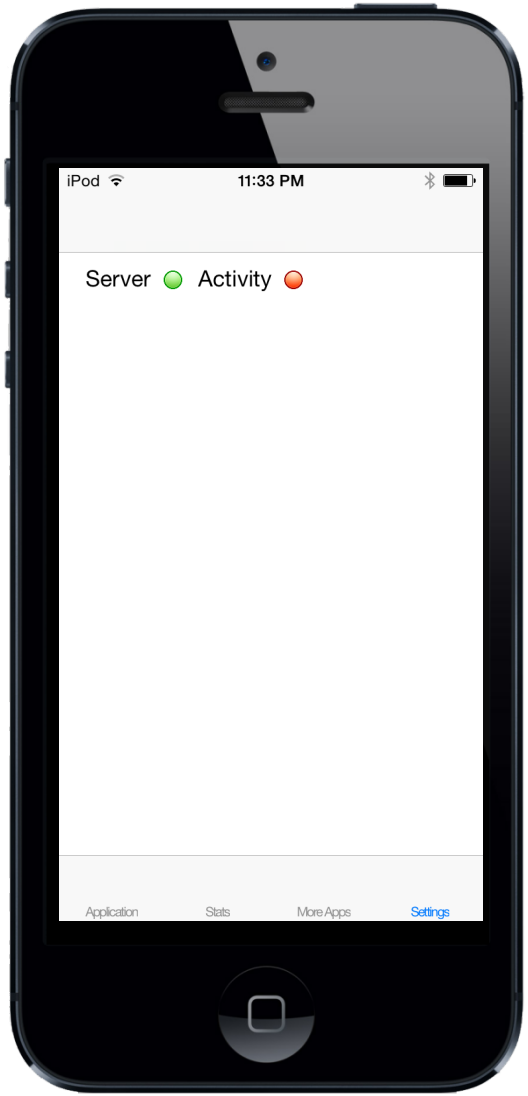


StepSense

- Easy integration into new prototypes
- Makes it simple to develop web applications for prototypes

Current Limitations

- Low latency not guaranteed
- Rigid configuration
- iOS limitations
  - Backgrounding
  - Identification



Future Work

- More gateway support
- Persistent connections
- Firmware integration / remote updates
- Longer-term studies

Thank You

Will McGrath
PhD Student
Stanford CS
wmcgrath@stanford.edu